Glossary

Abstract data type A data type whose properties (domain and operations) are specified independently of any particular implementation.

Abstract step A step for which some implementation details remain unspecified.

Abstraction barrier The invisible wall around a class object that encapsulates implementation details. The wall can be breached only through the public interface.

Aggregate operation An operation on a data structure as a whole, as opposed to an operation on an individual component of the data structure.

Algorithm A step-by-step procedure for solving a problem in a finite amount of time.

Anonymous type A type that does not have an associated type identifier.

Argument A variable or expression listed in a call to a function; also called *actual argument* or *actual parameter*.

Argument list A mechanism by which functions communicate with each other.

Arithmetic/logic unit (ALU) The component of the central processing unit that performs arithmetic and logical operations.

Array A collection of components, all of the same type, ordered on N dimensions ($N \geq 1$). Each component is accessed by N indexes, each of which represents the component's position within that dimension.

Assembler A program that translates an assembly language program into machine code.

Assembly language A low-level programming language in which a mnemonic is used to represent each of the machine language instructions for a particular computer.

Assignment expression A C++ expression with (1) a value and (2) the side effect of storing the expression value into a memory location.

Assignment statement A statement that stores the value of an expression into a variable.

Automatic variable A variable for which memory is allocated and deallocated when control enters and exits the block in which it is declared.
Auxiliary storage device A device that stores data in encoded form outside the computer's main memory.
Base address The memory address of the first element of an array.
Base case The case for which the solution can be stated nonrecursively.
Base class (superclass) The class being inherited from.
Binary operator An operator that has two operands.
Black box An electrical or mechanical device whose inner workings are hidden from view.
C string In C and C++, a null-terminated sequence of characters stored in a `char` array.
Catch To process a thrown exception. (The catching is performed by an exception handler.)
Central processing unit (CPU) The part of the computer that executes the instructions (program) stored in memory; made up of the arithmetic/logic unit and the control unit.
Class A structured type in a programming language that is used to represent an abstract data type.
Class member A component of a class. Class members may be either data or functions.
Class object (class instance) A variable of a **`class`** type.
Class template A C++ language construct that allows the compiler to generate multiple versions of a class by allowing parameterized data types.
Client Software that declares and manipulates objects of a particular class.
Communication complexity A measure of the quantity of data passing through a module's interface.
Compiler A program that translates a high-level language into machine code.
Complexity A measure of the effort expended by the computer in performing a computation, relative to the size of the computation.
Composition (containment) A mechanism by which the internal data (the state) of one class includes an object of another class.
Computer A programmable device that can store, retrieve, and process data.
Computer program A sequence of instructions to be performed by a computer.
Computer programming The process of planning a sequence of steps for a computer to follow.
Concrete step A step for which the implementation details are fully specified.
Constructor An operation that creates a new instance (variable) of an ADT.
Control abstraction The separation of the logical properties of an action from its implementation.
Control structure A statement used to alter the normally sequential flow of control.
Control unit The component of the central processing unit that controls the actions of the other components so that instructions (the program) are executed in the correct sequence.
Count-controlled loop A loop that executes a specified number of times.
Dangling pointer A pointer that points to a variable that has been deallocated.
Data Information in a form a computer can use.

Data abstraction The separation of a data type's logical properties from its implementation.
Data flow The flow of information from the calling code to a function and from the function back to the calling code.
Data representation The concrete form of data used to represent the abstract values of an abstract data type.
Data type A specific set of data values, along with a set of operations on those values.
Declaration A statement that associates an identifier with a data object, a function, or a data type so that the programmer can refer to that item by name.
Deep copy An operation that not only copies one class object to another but also makes copies of any pointed-to data.
Demotion (narrowing) The conversion of a value from a "higher" type to a "lower" type according to a programming language's precedence of data types. Demotion may cause loss of information.
Derived class (subclass) The class that inherits.
Direct addressing Accessing a variable in one step by using the variable name.
Documentation The written text and comments that make a program easier for others to understand, use, and modify.
Driver A simple `main` function that is used to call a function being tested. The use of a driver permits direct control of the testing process.
Dynamic binding The run-time determination of which function to call for a particular object.
Dynamic data Variables created during execution of a program by means of special operations. In C++, these operations are **`new`** and **`delete`**.
Dynamic data structure A data structure that can expand and contract during execution.
Dynamic linked list A linked list composed of dynamically allocated nodes that are linked together by pointers.
Editor An interactive program used to create and modify source programs or data.
Encapsulation Hiding a module implementation in a separate block with a formally specified interface.
Enumeration type A user-defined data type whose domain is an ordered set of literal values expressed as identifiers.
Enumerator One of the values in the domain of an enumeration type.
Evaluate To compute a new value by performing a specified set of operations on given values.
Event counter A variable that is incremented each time a particular event occurs.
Event-controlled loop A loop that terminates when something happens inside the loop body to signal that the loop should be exited.
Exception An unusual, often unpredicatable event, detectable by software or hardware, that requires special processing; also, in C++, a variable or class object that represents an exceptional event.
Exception handler A section of program code that is executed when a particular exception occurs.
Expression An arrangement of identifiers, literals, and operators that can be evaluated to compute a value of a given type.

Expression statement A statement formed by appending a semicolon to an expression.
External (head) pointer A pointer variable that points to the first node in a dynamic linked list.
External representation The printable (character) form of a data value.
Field (member, in C++) A component of a record.
File A named area in secondary storage that is used to hold a collection of data; the collection of data itself.
Flow of control The order in which the computer executes statements in a program.
Free store (heap) A pool of memory locations reserved for allocation and deallocation of dynamic data.
Function A subprogram in C++.
Function call (function invocation) The mechanism that transfers control to a function.
Function call (to a void function) A statement that transfers control to a void function. In C++, this statement is the name of the function, followed by a list of arguments.
Function definition A function declaration that includes the body of the function.
Function overloading The use of the same name for different C++ functions, distinguished from each other by their parameter lists.
Function prototype A function declaration without the body of the function.
Function template A C++ language construct that allows the compiler to generate multiple versions of a function by allowing parameterized data types.
Function value type The data type of the result value returned by a function.
Functional cohesion A property of a module in which all concrete steps are directed toward solving just one problem, and any significant subproblems are written as abstract steps. Also, the principle that a module should perform exactly one abstract action.
Functional decomposition A technique for developing software in which the problem is divided into more easily handled subproblems, the solutions of which create a solution to the overall problem.
Functional equivalence A property of a module that performs exactly the same operation as the abstract step it defines. A pair of modules are also functionally equivalent to each other when they perform exactly the same operation.
General case The case for which the solution is expressed in terms of a smaller version of itself; also known as *recursive case*.
Generic algorithm An algorithm in which the actions or steps are defined but the data types of the items being manipulated are not.
Generic data type A type for which the operations are defined but the data types of the items being manipulated are not.
Hardware The physical components of a computer.
Hierarchical record A record in which at least one of the components is itself a record.
Identifier A name associated with a function or data object and used to refer to that function or data object.
Inaccessible object A dynamic variable on the free store without any pointer pointing to it.

Indirect addressing Accessing a variable in two steps by first using a pointer that gives the location of the variable.
Infinite recursion The situation in which a function calls itself over and over endlessly.
Information Any knowledge that can be communicated.
Information hiding The encapsulation and hiding of implementation details to keep the user of an abstraction from depending on or incorrectly manipulating these details.
Inheritance A mechanism by which one class acquires the properties—the data and operations—of another class.
Input/output (I/O) devices The parts of the computer that accept data to be processed (input) and present the results of that processing (output).
Interactive system A system that allows direct communication between user and computer.
Interface A connecting link at a shared boundary that permits independent systems to meet and act on or communicate with each other. Also, the formal description of the purpose of a subprogram and the mechanism for communicating with it.
Internal representation The form in which a data value is stored inside the memory unit.
Iteration An individual pass through, or repetition of, the body of a loop.
Iteration counter A counter variable that is incremented with each iteration of a loop.
Iterator An operation that allows us to process—one at a time—all the components in an instance of an ADT.
Length The number of values currently stored in a list.
Lifetime The period of time during program execution when an identifier has memory allocated to it.
Linked list A list in which the order of the components is determined by an explicit link member in each node, rather than by the sequential order of the components in memory.
List A variable-length, linear collection of homogeneous components.
Literal value Any constant value written in a program.
Local variable A variable declared within a block and not accessible outside of that block.
Loop A control structure that causes a statement or group of statements to be executed repeatedly.
Loop entry The point at which the flow of control reaches the first statement inside a loop.
Loop exit The point at which the repetition of the loop body ends and control passes to the first statement following the loop.
Loop test The point at which the While expression is evaluated and the decision is made either to begin a new iteration or skip to the statement immediately following the loop.
Machine language The language, made up of binary-coded instructions, that is used directly by the computer.
Member selector The expression used to access components of a struct or class variable. It is formed by using the struct or class variable name and the member name, separated by a dot (period).

Memory leak The loss of available memory space that occurs when dynamic data is allocated but never deallocated.
Memory unit Internal data storage in a computer.
Metalanguage A language that is used to write the syntax rules for another language.
Mixed type expression An expression that contains operands of different data types; also called *mixed mode expression*.
Module A self-contained collection of steps that solves a problem or subproblem; can contain both concrete and abstract steps.
Name precedence The precedence that a local identifier in a function has over a global identifier with the same name in any references that the function makes to that identifier; also called *name hiding*.
Named constant (symbolic constant) A location in memory, referenced by an identifier, that contains a data value that cannot be changed.
Named type A user-defined type whose declaration includes a type identifier that gives a name to the type.
Nonlocal identifier With respect to a given block, any identifier declared outside that block.
Object-oriented design (OOD) A technique for developing software in which the solution is expressed in terms of objects—self-contained entities composed of data and operations on that data.
Object-oriented programming (OOP) The use of data abstraction, inheritance, and dynamic binding to construct programs that are collections of interacting objects.
Object program The machine language version of a source program.
Observer An operation that allows us to observe the state of an instance of an ADT without changing it.
One-dimensional array A structured collection of components, all of the same type, that is given a single name. Each component (array element) is accessed by an index that indicates the component's position within the collection.
Operating system A set of programs that manages all of the computer's resources.
Out-of-bounds array index An index value that, in C++, is either less than 0 or greater than the array size minus 1.
Parameter A variable declared in a function heading; also called *formal argument* or *formal parameter*.
Peripheral device An input, output, or auxiliary storage device attached to a computer.
Pointer type A simple data type consisting of an unbounded set of values, each of which addresses or otherwise indicates the location of a variable of a given type. Among the operations defined on pointer variables are assignment and testing for equality.
Polymorphic operation An operation that has multiple meanings depending on the type of the object to which it is bound at run time.
Postcondition An assertion that should be true after a module has executed.
Precision The maximum number of significant digits.
Precondition An assertion that must be true before a module begins executing.
Programming Planning or scheduling the performance of a task or an event.

Programming language A set of rules, symbols, and special words used to construct a computer program.
Promotion (widening) The conversion of a value from a "lower" type to a "higher" type according to a programming language's precedence of data types.
Range of values The interval within which values of a numeric type must fall, specified in terms of the largest and smallest allowable values.
Record (structure, in C++) A structured data type with a fixed number of components that are accessed by name. The components may be heterogeneous (of different types).
Recursive algorithm A solution that is expressed in terms of (a) smaller instances of itself and (b) a base case.
Recursive call A function call in which the function being called is the same as the one making the call.
Recursive definition A definition in which something is defined in terms of smaller versions of itself.
Reference parameter A parameter that receives the location (memory address) of the caller's argument.
Reference type A simple data type consisting of an unbounded set of values, each of which is the address of a variable of a given type. The only operation defined on a reference variable is initialization, after which every appearance of the variable is implicitly dereferenced.
Representational error Arithmetic error that occurs when the precision of the true result of an arithmetic operation is greater than the precision of the machine.
Reserved word A word that has special meaning in C++; it cannot be used as a programmer-defined identifier.
Scope The region of program code where it is legal to reference (use) an identifier.
Scope rules The rules that determine where in the program an identifier may be accessed, given the point where that identifier is declared.
Self-documenting code Program code containing meaningful identifiers as well as judiciously used clarifying comments.
Semantics The set of rules that determines the meaning of instructions written in a programming language.
Shallow copy An operation that copies one class object to another without copying any pointed-to data.
Short-circuit (conditional) evaluation Evaluation of a logical expression in left-to-right order with evaluation stopping as soon as the final truth value can be determined.
Side effect Any effect of one function on another that is not a part of the explicitly defined interface between them.
Significant digits Those digits from the first nonzero digit on the left to the last nonzero digit on the right (plus any 0 digits that are exact).
Simple (atomic) data type A data type in which each value is atomic (indivisible).
Software Computer programs; the set of all programs available on a computer.
Software engineering The application of traditional engineering methodologies and techniques to the development of software.
Software piracy The unauthorized copying of software for either personal use or use by others.

Sorting Arranging the components of a list into order (for instance, words into alphabetical order or numbers into ascending or descending order).
Source program A program written in a high-level programming language.
Static binding The compile-time determination of which function to call for a particular object.
Static variable A variable for which memory remains allocated throughout the execution of the entire program.
Structured data type A data type in which each value is a collection of components and whose organization is characterized by the method used to access individual components. The allowable operations on a structured data type include the storage and retrieval of individual components.
Structured (procedural) programming The construction of programs that are collections of interacting functions or procedures.
Stub A dummy function that assists in testing part of a program. A stub has the same name and interface as a function that actually would be called by the part of the program being tested, but it is usually much simpler.
Switch expression The expression whose value determines which switch label is selected. It cannot be a floating-point or string expression.
Syntax The formal rules governing how valid instructions are written in a programming language.
Tail recursion A recursive algorithm in which no statements are executed after the return from the recursive call.
Termination condition The condition that causes a loop to be exited.
Test plan A document that specifies how a program is to be tested.
Test plan implementation Using the test cases specified in a test plan to verify that a program outputs the predicted results.
Testing the state of a stream The act of using a C++ stream object in a logical expression as if it were a Boolean variable; the result is `true` if the last I/O operation on that stream succeeded, and `false` otherwise.
Throw To signal the fact that an exception has occurred; also called *raise*.
Transformer An operation that builds a new value of the ADT, given one or more previous values of the type.
Two-dimensional array A collection of components, all of the same type, structured in two dimensions. Each component is accessed by a pair of indexes that represent the component's position in each dimension.
Type casting The explicit conversion of a value from one data type to another; also called type conversion.
Type coercion The implicit (automatic) conversion of a value from one data type to another.
Unary operator An operator that has just one operand.
Value parameter A parameter that receives a copy of the value of the corresponding argument.
Value-returning function A function that returns a single value to its caller and is invoked from within an expression.
Variable A location in memory, referenced by an identifier, that contains a data value that can be changed.

Virus A computer program that replicates itself, often with the goal of spreading to other computers without authorization, and possibly with the intent of doing harm.

Void function (procedure) A function that does not return a function value to its caller and is invoked as a separate statement.